Spacework

Project Conception Document

This project has been created as part of the $ft_{t}rancendance$.

Document Version: 5.0
Timeline: 6 Weeks (1.5 Months)
Methodology: Agile Scrum

December 2025

Contents

1 Description **2**
- 1.1 Project Goal . . . 2
- 1.2 Key Features . . . 2

2 Agile Methodology **2**
- 2.1 Scrum Framework . . . 2
  - 2.1.1 Roles . . . 2
  - 2.1.2 Ceremonies . . . 2
- 2.2 Sprint Planning (6 Weeks / 1.5 Months) . . . 3
- 2.3 Story Points and Estimation . . . 3

3 User Types and Roles **3**
- 3.1 System Roles (Organization Level) . . . 3
- 3.2 Role Permissions Matrix . . . 4

4 Class Diagram **4**

5 User Flows **6**
- 5.1 Authentication Flow . . . 6
- 5.2 Main Application Navigation . . . 6
- 5.3 Task Management Flow . . . 7
- 5.4 Virtual Office Flow . . . 8
- 5.5 Gamification Flow . . . 9

6 Pages and Screens for Figma Design **9**
- 6.1 Page List . . . 9
  - 6.1.1 Authentication Pages . . . 9
  - 6.1.2 Dashboard Pages . . . 10
  - 6.1.3 Gamification Pages . . . 10
  - 6.1.4 Virtual Office Pages . . . 10
  - 6.1.5 User and Organization Pages . . . 10
  - 6.1.6 Additional Pages . . . 10
- 6.2 Component Library . . . 11
  - 6.2.1 Core Components . . . 11
  - 6.2.2 Gamification Components . . . 11
  - 6.2.3 Project Components . . . 11

7 Technical Stack **12**
- 7.1 Frontend . . . 12
- 7.2 Backend . . . 12
- 7.3 DevOps . . . 12

8 Database Schema (Prisma) **12**

1 Description

Spacework is a gamified team productivity and collaboration platform that transforms remote work into an engaging, game-like experience. The platform combines project management, real-time collaboration, and gamification elements (XP, levels, achievements, quests) with a virtual office environment where team members can interact spatially.

1.1 Project Goal

Create a comprehensive web application that demonstrates technical proficiency across multiple domains while providing a unique and engaging user experience for remote teams.

1.2 Key Features

- **Virtual Office:** Interactive 2D workspace with themed rooms
- **Real-time Collaboration:** Live chat, presence, and notifications
- **Gamification System:** XP, levels, achievements, quests, leaderboards
- **Project Management:** Kanban boards, task tracking, sprints
- **Role-Based Access:** Multiple user types with specific permissions
- **Rewards System:** Redeem XP for recognition

2 Agile Methodology

2.1 Scrum Framework

We follow the **Agile Scrum** methodology for project development.

2.1.1 Roles

Role	Responsibility
Product Owner	Defines product vision, manages backlog, prioritizes features
Scrum Master	Facilitates ceremonies, removes blockers, ensures process adherence
Development Team	Implements features, participates in all ceremonies

Table 1: Scrum team roles

2.1.2 Ceremonies

Ceremony	Frequency	Duration	Purpose
Sprint Planning	Start of sprint	1 hour	Plan sprint backlog
Daily Standup	Daily	15 min	Sync progress, identify blockers
Sprint Review	End of sprint	30 min	Demo completed work
Sprint Retrospective	End of sprint	30 min	Improve process

Table 2: Scrum ceremonies

2.2 Sprint Planning (6 Weeks / 1.5 Months)

Sprint Duration: 1 Week

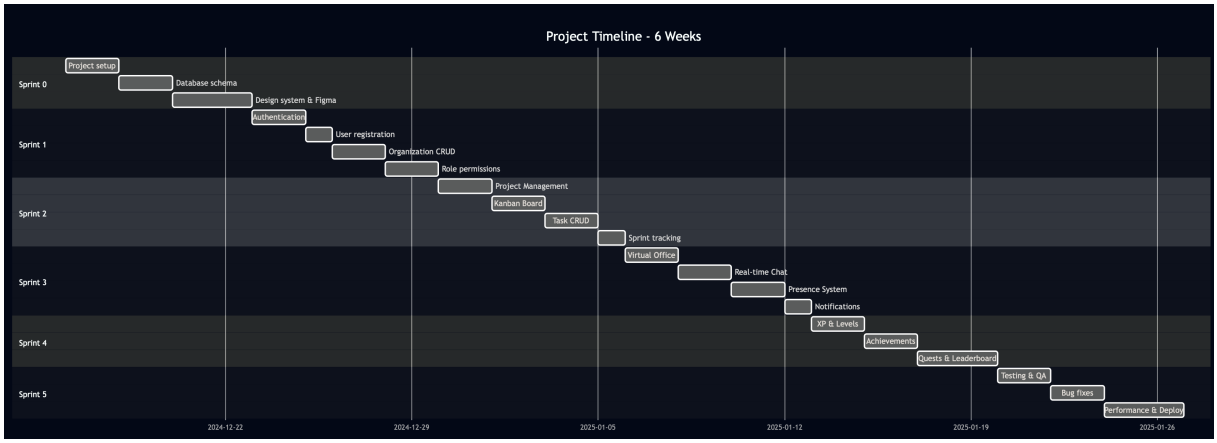


Figure 1: Project timeline (6 weeks)

2.3 Story Points and Estimation

Points	Complexity	Time Estimate
1	Trivial	Less than 2 hours
2	Simple	2–4 hours
3	Medium	4–8 hours
5	Complex	1–2 days
8	Very Complex	2–3 days

Table 3: Story points estimation

3 User Types and Roles

3.1 System Roles (Organization Level)

Role	Description	Permissions
Owner	Creates and owns organization	Full organization control, billing, delete organization
Admin	Manages organization	Manage members, projects, settings
Member	Regular organization member	Access assigned projects and tasks

Table 4: Organization roles

3.2 Role Permissions Matrix

Permission	Owner	Admin	Member
Manage Organization	Yes	Yes	No
Invite/Remove Members	Yes	Yes	No
Assign Roles	Yes	Yes	No
Create Projects	Yes	Yes	No
Delete Projects	Yes	Yes	No
Manage Sprints	Yes	Yes	No
Create Tasks	Yes	Yes	Yes
Assign Tasks	Yes	Yes	No
Edit Any Task	Yes	Yes	No
Edit Own Tasks	Yes	Yes	Yes
View Reports	Yes	Yes	No
Access Virtual Office	Yes	Yes	Yes
Use Chat	Yes	Yes	Yes
Earn XP	Yes	Yes	Yes

Table 5: Role permissions matrix

4 Class Diagram

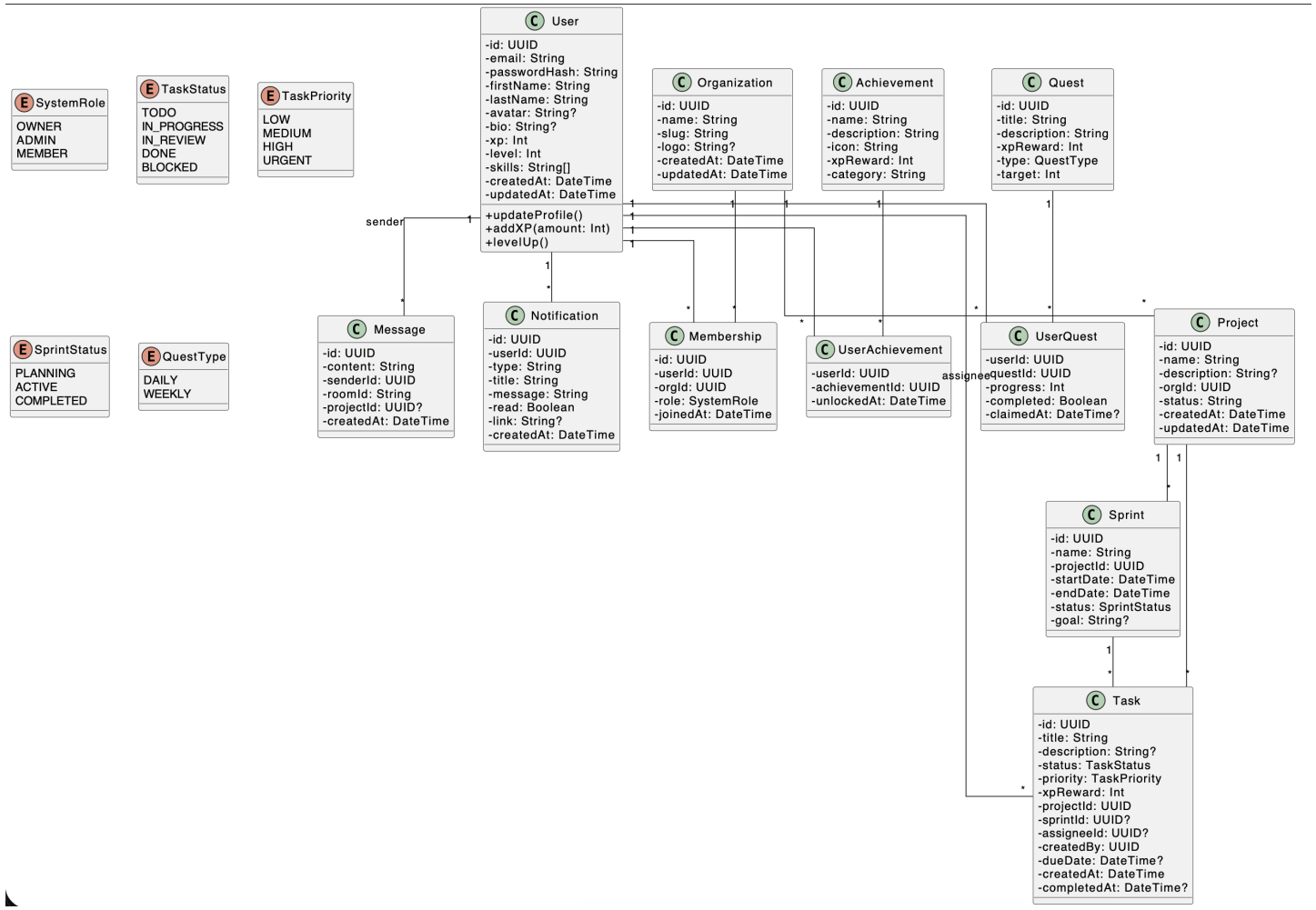


Figure 2: UML class diagram

5 User Flows

5.1 Authentication Flow

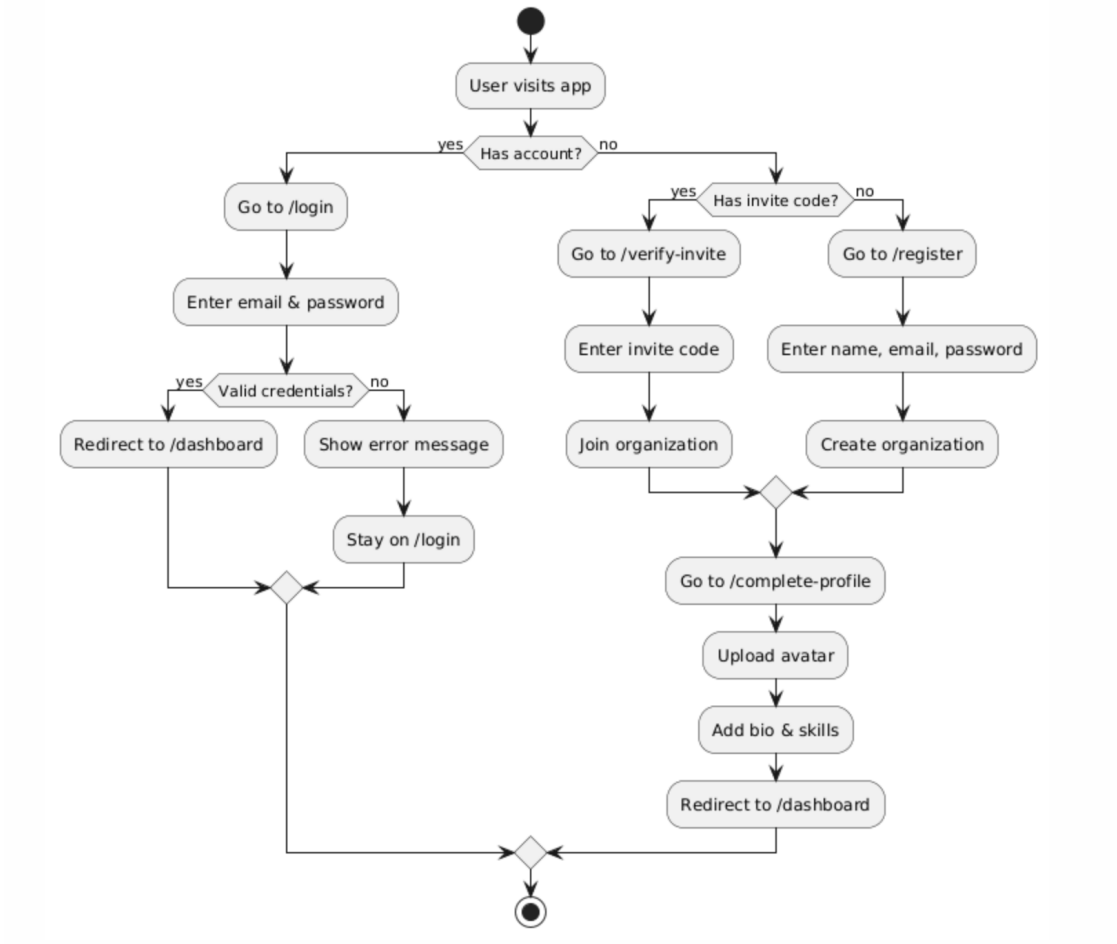


Figure 3: Authentication flow

5.2 Main Application Navigation

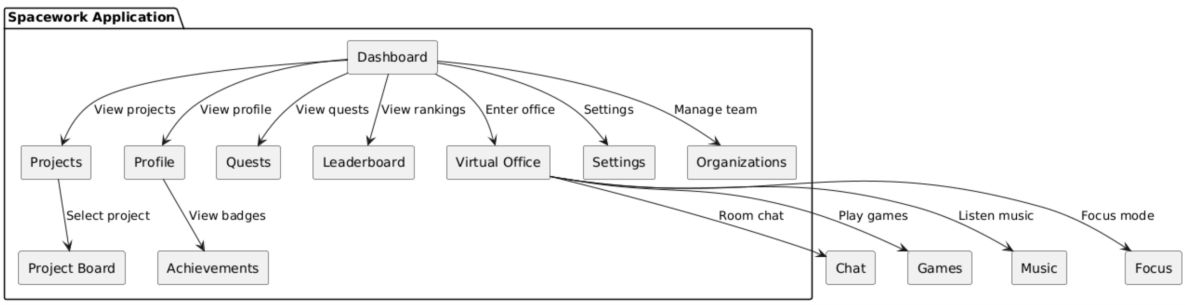
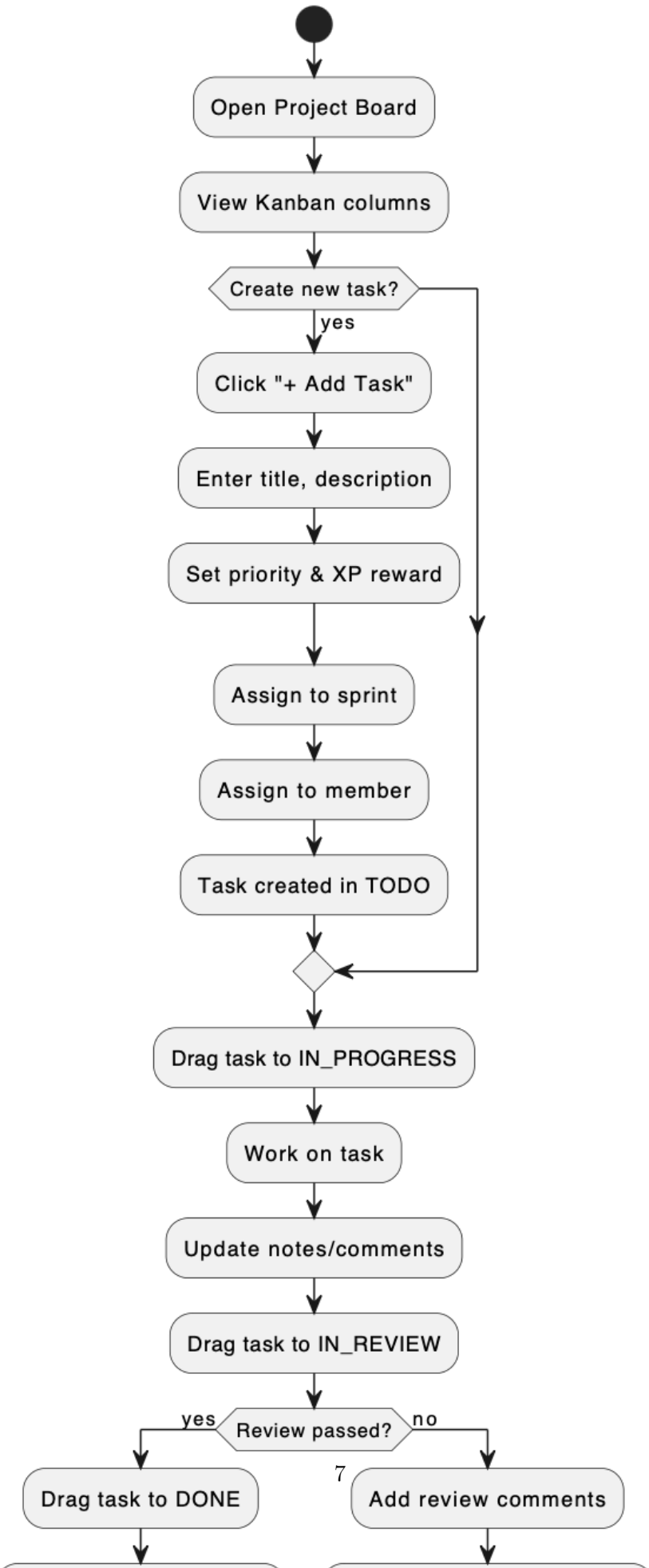


Figure 4: Main application navigation flow

5.3 Task Management Flow



5.4 Virtual Office Flow

Figure 6: Virtual office flow

5.5 Gamification Flow

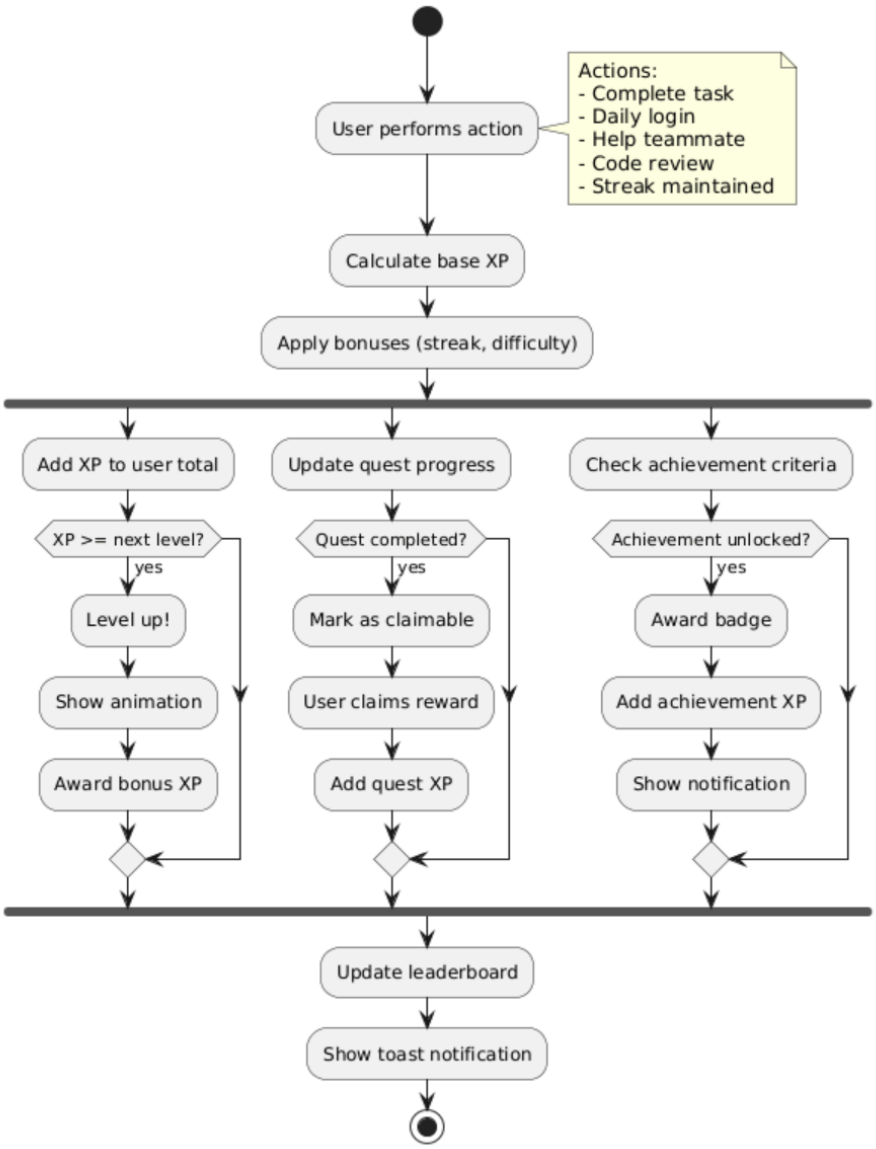


Figure 7: Gamification flow

6 Pages and Screens for Figma Design

6.1 Page List

6.1.1 Authentication Pages

Table 6: Authentication pages

Page	Route	Priority
Login	/login	High
Register	/register	High
Verify Invite	/verify-invite	High
Complete Profile	/complete-profile	High

Page	Route	Priority
Forgot Password	/forgot-password	Medium

6.1.2 Dashboard Pages

Table 7: Dashboard pages

Page	Route	Priority
Dashboard Home	/dashboard	High
Projects List	/projects	High
Project Board (Kanban)	/projects/[id]	High
Task Detail Modal	Modal	High

6.1.3 Gamification Pages

Table 8: Gamification pages

Page	Route	Priority
Quests	/quests	High
Achievements	/achievements	Medium
Leaderboard	/leaderboard	High

6.1.4 Virtual Office Pages

Table 9: Virtual office pages

Page	Route	Priority
Virtual Office	/playground	High
Room Activities Panel	Side Panel	High

6.1.5 User and Organization Pages

Table 10: User and organization pages

Page	Route	Priority
Profile	/profile	High
Settings	/settings	Medium
Organizations	/organizations	High

6.1.6 Additional Pages

Table 11: Additional pages

Page	Route	Priority
Activity Feed	/activity	Low
Focus Timer	/focus	Low
Chat	/chat	Medium

6.2 Component Library

6.2.1 Core Components

Table 12: Core components

Component	Variants
Button	Primary, Secondary, Outline, Ghost, Destructive
Input	Text, Email, Password, Search
Card	Default, Interactive, Highlighted
Badge	Default, Outline, Role colors
Avatar	XS, SM, MD, LG, XL; Online indicator
Dialog	Default, Alert, Form
Dropdown	Menu, Select
Tabs	Default
Progress	Bar, Circle
Toast	Success, Error, Warning, Info

6.2.2 Gamification Components

Table 13: Gamification components

Component	Description
XP Progress Bar	Level progress with animation
Achievement Card	Badge with locked/unlocked states
Quest Card	Progress, reward, claim button
Leaderboard Row	Rank, avatar, name, XP
Level Badge	Current level display
Streak Counter	Streak counter with icon and count

6.2.3 Project Components

Table 14: Project management components

Component	Description
Task Card	Title, priority, assignee, XP
Kanban Column	Header, task list, add button
Sprint Selector	Dropdown with sprint info
Project Card	Name, progress, team avatars

7 Technical Stack

7.1 Frontend

Technology	Purpose
Next.js	React framework with App Router
TypeScript	Type-safe development
Tailwind CSS	Utility-first styling
shadcn/ui	UI component library
Zustand	State management
React Hook Form	Form handling
Zod	Schema validation
Recharts	Data visualization
Socket.io Client	Real-time communication

Table 15: Frontend technology stack

7.2 Backend

Technology	Purpose
Next.js API Routes	REST API endpoints
PostgreSQL	Relational database
Prisma	ORM
Socket.io	WebSocket server
bcrypt	Password hashing

Table 16: Backend technology stack

7.3 DevOps

Technology	Purpose
Docker	Containerization
Docker Compose	Multi-container orchestration

Table 17: DevOps technology stack

8 Database Schema (Prisma)

Listing 1: Prisma schema (schema.prisma)

```
generator client {
  provider = "prisma-client-js"
}

datasource db {
  provider = "postgresql"
  url      = env("DATABASE_URL")
}

```

```
enum SystemRole {
  OWNER
  ADMIN
  MEMBER
}

enum TaskStatus {
  TODO
  IN_PROGRESS
  IN_REVIEW
  DONE
  BLOCKED
}

enum TaskPriority {
  LOW
  MEDIUM
  HIGH
  URGENT
}

enum SprintStatus {
  PLANNING
  ACTIVE
  COMPLETED
}

enum QuestType {
  DAILY
  WEEKLY
}

model User {
  id           String   @id @default(uuid())
  email        String   @unique
  passwordHash String
  firstName    String
  lastName     String
  avatar       String?
  bio          String?
  xp           Int      @default(0)
  level        Int      @default(1)
  skills       String[]
  createdAt    DateTime @default(now())
  updatedAt    DateTime @updatedAt

  memberships      Membership[]
  assignedTasks    Task[]           @relation("TaskAssignee")
  createdTasks     Task[]           @relation("TaskCreator")
  messages         Message[]
  achievements     UserAchievement[]
  quests           UserQuest[]
  notifications    Notification[]
}

model Organization {
  id        String   @id @default(uuid())
  name      String
```

```
  slug      String   @unique
  logo      String?
  createdAt DateTime @default(now())
  updatedAt DateTime @updatedAt

  memberships Membership[]
  projects    Project[]
}

model Membership {
  id        String     @id @default(uuid())
  userId    String
  orgId     String
  role      SystemRole @default(MEMBER)
  joinedAt  DateTime   @default(now())

  user         User         @relation(fields: [userId], references: [id])
  organization Organization @relation(fields: [orgId], references: [id])

  @@unique([userId, orgId])
}

model Project {
  id          String   @id @default(uuid())
  name        String
  description String?
  orgId       String
  status      String   @default("active")
  createdAt   DateTime @default(now())
  updatedAt   DateTime @updatedAt

  organization Organization @relation(fields: [orgId], references: [id])
  sprints      Sprint[]
  tasks        Task[]
  messages     Message[]
}

model Sprint {
  id        String       @id @default(uuid())
  name      String
  projectId String
  startDate DateTime
  endDate   DateTime
  status    SprintStatus @default(PLANNING)
  goal      String?

  project Project @relation(fields: [projectId], references: [id])
  tasks   Task[]
}

model Task {
  id          String       @id @default(uuid())
  title       String
  description String?
  status      TaskStatus   @default(TODO)
  priority    TaskPriority @default(MEDIUM)
  xpReward    Int          @default(10)
  projectId   String
```

```
  sprintId    String?
  assigneeId  String?
  createdBy   String
  dueDate     DateTime?
  createdAt   DateTime     @default(now())
  completedAt DateTime?

  project  Project @relation(fields: [projectId], references: [id])
  sprint   Sprint? @relation(fields: [sprintId], references: [id])
  assignee User?   @relation("TaskAssignee", fields: [assigneeId], references: [id
    ])
  creator  User    @relation("TaskCreator", fields: [createdBy], references: [id])
}

model Message {
  id        String   @id @default(uuid())
  content   String
  senderId  String
  roomId    String
  projectId String?
  createdAt DateTime @default(now())

  sender  User     @relation(fields: [senderId], references: [id])
  project Project? @relation(fields: [projectId], references: [id])
}

model Achievement {
  id          String @id @default(uuid())
  name        String
  description String
  icon        String
  xpReward    Int
  category    String

  users UserAchievement[]
}

model UserAchievement {
  userId        String
  achievementId String
  unlockedAt    DateTime @default(now())

  user        User        @relation(fields: [userId], references: [id])
  achievement Achievement @relation(fields: [achievementId], references: [id])

  @@id([userId, achievementId])
}

model Quest {
  id          String    @id @default(uuid())
  title       String
  description String
  xpReward    Int
  type        QuestType
  target      Int

  users UserQuest[]
}
```

```

model UserQuest {
  userId    String
  questId   String
  progress  Int      @default(0)
  completed Boolean  @default(false)
  claimedAt DateTime?

  user  User  @relation(fields: [userId], references: [id])
  quest Quest @relation(fields: [questId], references: [id])

  @@id([userId, questId])
}

model Notification {
  id        String   @id @default(uuid())
  userId    String
  type      String
  title     String
  message   String
  read      Boolean  @default(false)
  link      String?
  createdAt DateTime @default(now())

  user User @relation(fields: [userId], references: [id])
}
```

End of Document